Entropy Regularization and Temperature Scaling in Large Language Models

Introduction

Large language models (LLMs) are designed to predict token probabilities based on the context provided in the input sequence. These models generate coherent and contextually relevant text by optimizing their prediction accuracy. The training of LLMs typically involves the **cross-entropy (CE) loss function**, which minimizes the difference between predicted probabilities and true token distributions.

During inference (after training), **temperature scaling** is applied to adjust the output probabilities, making the model less overconfident and more exploratory. This is particularly useful for generating diverse outputs and preventing the model from making deterministic, repetitive predictions. The key challenge, however, lies in understanding how **entropy regularization** during training can replicate the effects of temperature scaling during inference, as both techniques aim to reduce overconfidence and increase exploration.

Cross-Entropy Loss Function in LLM Training

The cross-entropy loss function is defined as:

$$L_{\text{CE}} = -\sum_{i=1}^{V} P(i) \log Q(i)$$

Where:

- $P(i)$ is the true probability of token i,
- $Q(i)$ is the predicted probability of token i,
- V is the number of possible tokens.

The goal of this loss function is to minimize the distance between the predicted probabilities Q and the true probabilities P. When P is sparse (i.e., most of the tokens have zero probability except for a few), the model tends to output highly **peaky** distributions, which concentrate probability mass on just a few tokens. This is desirable for improving prediction accuracy, but can lead to overfitting and lack of diversity in the generated output.

Temperature Scaling and Its Impact

After training, **temperature scaling** is applied to soften the model's predictions, making it less confident. Temperature scaling modifies the logits z (the raw model outputs) by dividing them by a temperature parameter T:

$$Q(i) = \frac{\exp(z_i/T)}{\sum_{j=1}^{V} \exp(z_j/T)}$$

Where T is a hyperparameter:

- If $T > 1$, the distribution becomes **smoother** (i.e., more uniform), making the model less confident in its predictions,

- If $T = 1$, no scaling occurs, and the model's raw logits are used,

- If $T < 1$, the model becomes more **deterministic**, concentrating more probability on fewer tokens.

The result of temperature scaling is to reduce overconfidence, making the model more exploratory by flattening the distribution over all possible tokens.

Entropy Regularization and Its Role in Training

Entropy regularization introduces an additional term to the training loss function that penalizes highly deterministic predictions, encouraging the model to **spread** its probability mass more evenly. The regularized loss function becomes:

$$L_{\text{total}} = L_{\text{CE}} + \lambda H(Q)$$

Where $H(Q)$ is the **entropy** of the predicted distribution Q:

$$H(Q) = -\sum_{i=1}^{V} Q(i) \log Q(i)$$

And λ is a hyperparameter that controls the strength of the regularization. The entropy term increases the model's entropy, pushing the distribution towards a more uniform one, and thereby making the model less confident in its predictions.

This regularization ensures that the model does not output overly peaky distributions, which could result in overfitting and lack of generalization. It encourages the model to explore different token possibilities rather than relying on a single ”best” token.

Demonstrating the Equivalence Between Entropy Regularization and Temperature Scaling

The core idea behind both entropy regularization and temperature scaling is to **reduce overconfidence** by making the token probabilities more spread out. Here's how they compare:

- **Temperature Scaling**: Reduces overconfidence by softening the logits, which smoothes the output probabilities. This is achieved during inference, after the model has been trained.
- **Entropy Regularization**: Directly influences the training process by penalizing deterministic, overconfident outputs through the entropy term $H(Q)$. The regularization encourages smoother distributions even during training.

Equivalence: Both techniques push the model's predictions to be less deterministic, i.e., they make the model avoid assigning excessive probability mass to a single token. This results in higher entropy in the predicted distribution. Temperature scaling achieves this smoothing during inference by modifying the logits, whereas entropy regularization achieves it during training by modifying the objective function.

Mathematical Derivation of the Equivalence

1. **Temperature Scaling** softens the logits during inference:

$$Q(i) = \frac{\exp(z_i/T)}{\sum_{j=1}^{V} \exp(z_j/T)}$$

2. **Entropy Regularization** during training encourages the model to spread out the probability distribution, increasing entropy $H(Q)$:

$$L_{\text{total}} = L_{\text{CE}} + \lambda H(Q)$$

By appropriately tuning λ, we can induce a similar smoothing effect as temperature scaling. The regularization term $H(Q)$ effectively encourages the model to avoid sharp, overconfident distributions, which is the primary effect of temperature scaling.

Conclusion

Both entropy regularization and temperature scaling serve to **reduce model overconfidence** and encourage more exploratory behavior by smoothing the token probability distributions. While temperature scaling achieves this effect post-training by modifying the logits, entropy regularization influences the model's behavior during training by adding a penalty for overly deterministic predictions. The equivalence between these two methods lies in their ability to increase entropy and spread out the probability mass over all tokens, ultimately leading to more diverse and generalized predictions.

By understanding and leveraging both techniques, LLMs can be trained to produce more robust, generalizable models that generate more diverse and creative outputs, mitigating the risk of overfitting or repetitive predictions.